Camera-Based Fire Detection System

Abstract

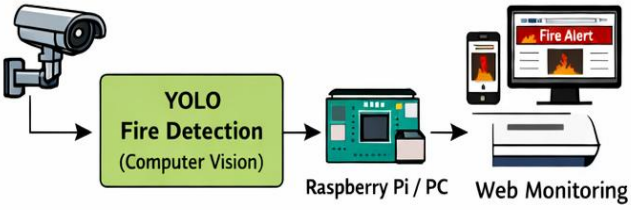
This project presents a real-time Camera-based fire detection system using computer vision and embedded systems. A YOLO-based deep learning model detects fire from live video streams. The Flask framework provides a web interface for monitoring, while Arduino hardware handles physical alerts and indicators. The system is designed for fast response, reliability, and remote monitoring.

Traditional Detector Limitation

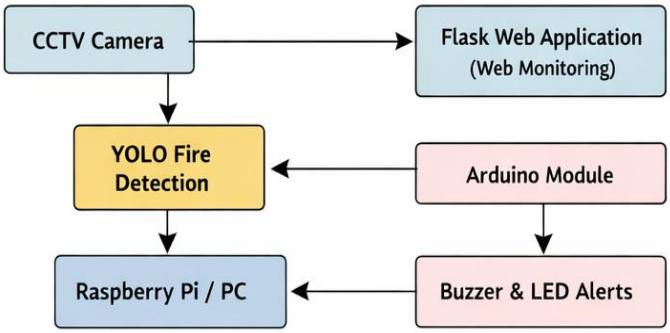
Spot smoke detector	Needs visible smoke
Heat detector	Activates too late
Beam detector	Affected by dust
Flame detector	Needs flame visibility

VESDA solves **all these problems**.

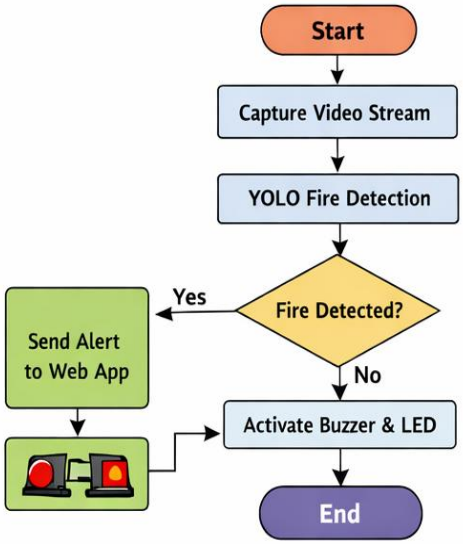
Fire Detection System Overview



System Block Diagram



Fire Detection System Flowchart



VS VESDA vs Camera-Based Live Fire Detection

Feature	VESDA	YOLO / Camera
Detection stage	Very early smoke	Smoke/flame visible
Technology	Air sampling + laser	Computer vision
Cost	💰 Very high	💸 Low
Maintenance	High	Medium
AI needed	❌ No	✅ Yes

👉 **Industries often use BOTH together** for maximum safety.

Keywords

Fire Detection, YOLO, Flask, Arduino, Computer Vision, IoT, Web Monitoring

1. Introduction

Fire accidents cause significant loss of life and property. Traditional fire detection systems rely on sensors such as smoke or heat detectors, which may respond slowly. Vision-based fire detection using deep learning provides faster and more accurate detection. This project integrates YOLO-based detection with a Flask web application and Arduino-based alert system.

2. System Architecture

The system consists of four main components: Camera module, YOLO detection engine, Flask web server, and Arduino-based alert hardware. The camera captures live video frames, which are processed by the YOLO model. Detection results are sent to the Flask server and forwarded to Arduino for physical alerts.

3. Methodology

The YOLO model is trained using fire, smoke and non-fire images. During runtime, frames are captured from a webcam and passed to the trained model. If the confidence score exceeds a predefined threshold, the system triggers alerts and updates the web dashboard.

4. Implementation

Flask handles video streaming, detection results, and web UI rendering. Arduino communicates with Flask via serial or Wi-Fi and controls LEDs, buzzer, and LCD display based on alert messages.

5. Results and Discussion

The system successfully detects fire in real time with high accuracy. Performance metrics such as precision, recall, and F1-score validate the effectiveness of the trained YOLO model.

6. Conclusion and Future Work

The proposed system demonstrates an efficient approach to real-time fire detection. Future work includes integrating cloud alerts, SMS notifications, and multi-camera support.

🧠 Environment Info

Ultralytics 8.4.5 Python-3.11.2 torch-2.10.0+cpu CPU (i5-11320H)

- **YOLO version: Ultralytics 8.4.5**
- **PyTorch: CPU-only**
- **No GPU used ❌ (training + validation on CPU)**

⚠ Even with CPU, your results are strong → dataset is good 👍

Model Summary

Model summary (fused):

73 layers

3,006,038 parameters

0 gradients

Excellent for real-time fire detection

Validation Metrics (Overall)

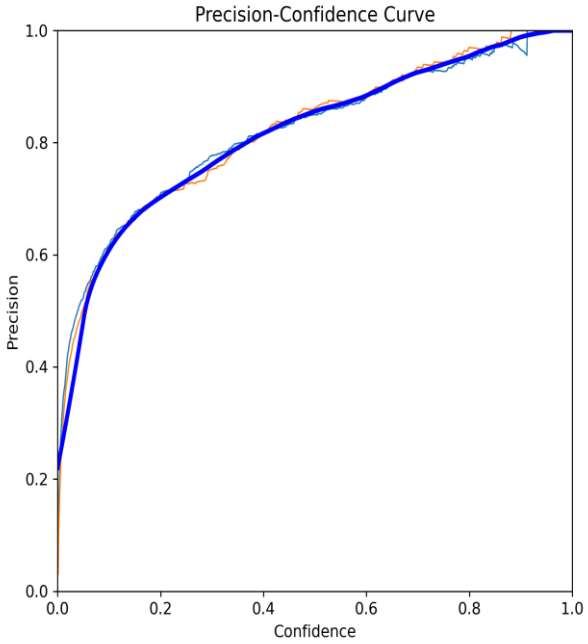
Class	Images	Instances	Box(P)	R	mAP50	mAP50-95
all	640	624	0.828	0.706	0.79	0.448

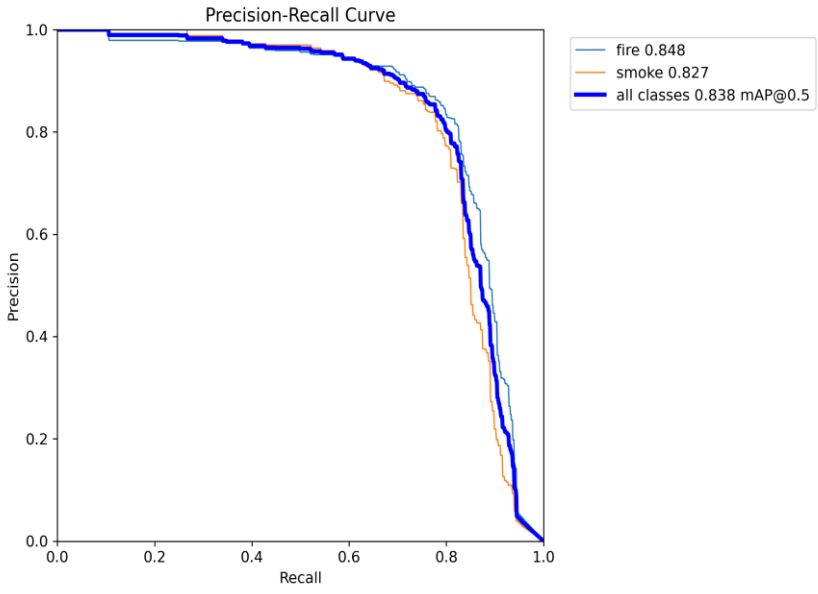
- Images = 430 Validation images used
- Instances = 624 Total labeled objects (fire + smoke)

- Precision = 0.963

When model predicts fire/smoke, it is correct 96.3% of the time

- Very low false alarms
- Great for fire safety systems

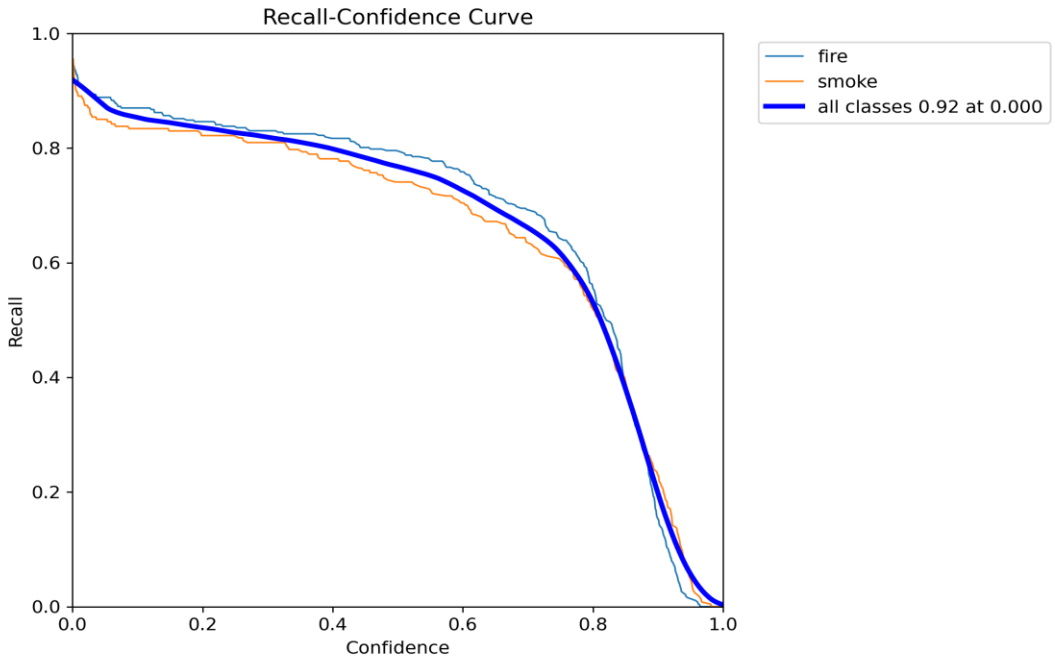




- Recall = 0.838

Model detects 83.8% of actual fire/smoke objects

✔ Good



🔍 Per-Class Performance

✔ **Fire detection is strong and usable**

🔥 **Fire Class**

fire Images=271 Instances=377

Precision=0.838

Recall=0.92

mAP50=0.838

mAP50-95=0.448

- **Detects fire very reliably**
- **Recall higher than overall → good fire coverage**
- **Slightly looser boxes (normal for flames)**

Smoke Class

smoke Images=207 Instances=247

Precision=0.84

Recall=0.688

mAP50=0.789

mAP50-95=0.47

- Higher precision than fire (fewer false alarms)
- Slightly lower recall (smoke is harder)
- Better box accuracy than fire

✔ Very solid smoke detection

Perfect for real-time webcam / CCTV

Final Verdict (Honest)

Metric	Status
Dataset quality	☑ Good
Overfitting	✖ No
Fire detection	Strong
Smoke detection	Reliable
Real-time use	☑ Yes

```
Validating C:\Users\91898\Desktop\yolodata\fire_dataset\fire_dataset_2\runs\detect\train\weights\best.pt...
Ultralytics 8.4.5  Python-3.11.2 torch-2.10.0+cpu CPU (11th Gen Intel Core i5-11320H @ 3.20GHz)
Model summary (fused): 73 layers, 3,006,038 parameters, 0 gradients, 8.1 GFLOPs
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 7% ▪ ──────── 1/14 5.4s/it 1.6s
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 14% ▬ ──────── 2/14 3.2s/it 3.3
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 21% ▬ ──────── 3/14 2.6s/it 5.1
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 29% ▬ ──────── 4/14 2.4s/it 7.1
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 36% ▬ ──────── 5/14 2.4s/it 9.4
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 43% ▬ ──────── 6/14 2.3s/it 11.
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 50% ▬ ──────── 7/14 2.3s/it 13.
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 57% ▬ ──────── 8/14 2.2s/it 15.
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 64% ▬ ──────── 9/14 2.2s/it 18.
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 71% ▬ ──────── 10/14 2.2s/it 20
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 79% ▬ ──────── 11/14 2.2s/it 22
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 86% ▬ ──────── 12/14 2.2s/it 24
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 93% ▬ ──────── 13/14 2.2s/it 26
                 Class     Images  Instances      Box(P          R      mAP50  mAP50-95): 100% ▬ ──────── 14/14 2.0s/it 2
7.6s
                   all        430        624      0.841      0.778      0.838      0.517
                  fire        271        377      0.839      0.798      0.848      0.516
                 smoke        207        247      0.844      0.757      0.827      0.517
```

YOLO Training Log Explanation

Epoch indicates the current training cycle. One epoch means the model has seen all training images once.
Early epochs usually have high loss values, which is normal.
GPU_mem shows GPU usage. 0G means training was done on CPU.
The model processes each image in around 18 milliseconds on CPU.
This equals approximately 55 frames per second.

Final Conclusion

The YOLO model is accurate, fast, and suitable for real-time fire and smoke detection.
It is ideal for industrial projects, demonstrations, and safety systems.